My CTF write-up

BY OLENA VERBYTSKA,

FACULTY OF MECHANICS AND MATHEMATICS OF

TARAS SHEVCHENKO NATIONAL UNIVERSITY OF KYIV STUDENT

Contents

Simple Crypto 1

- It was enough to google this string and I saw the answer. MD5 hash and well-known phrase.

- https://md5hashing.net/hash/md5/fc5e038d38a57032085441e7fe7010b0

- CTF_FLAG{helloworld}

Simple Crypto 2

- That's pretty easy to recognize Base64 format, so…

- https://www.boxentriq.com/code-breaking/base64-decoder

- CTF_FLAG{th1s_one_was_pretty_easy}

Simple Crypto 3

- Just removed "%" and converted hex to ASCII:

```
c = '%4e%6a%45%32%5a%54%4d%77%4e%7a%51%32%4f%44%59\
    %31%4e%7a%49%31%5a%6a%63%30%4e%6a%45%33%4d%7a%5a%69'
# n = c.split('%')
n = ['4e', '6a', '45', '32', '5a', '54', '4d', '77',
     '4e', '7a', '51', '32', '4f', '44', '59', '31',
     '4e', '7a', '49', '31', '5a', '6a', '63', '30',
     '4e', '6a', '45', '33', '4d', '7a', '5a', '69']
hex_n = ['0x' + i for i in n]
int_n = [int(i, 16) for i in hex_n]
msg = [chr(i) for i in int_n]
msg = ''.join(msg)
print(msg)
# msg = 'NjE2ZTMwNzQ2ODY1NzI1Zjc0NjE3MzZi'
```

- Then using Base64 decoder: msg => "616e30746865725f7461736b".
- And once again: hex to ASCII.

- CTF_FLAG{an0ther_task}

Simple Crypto 4

- The string was really huge and seemed to have Base64 format. Well I decoded this more than 20 times (probably 23), until obtained readable word.

- https://www.base64decode.org

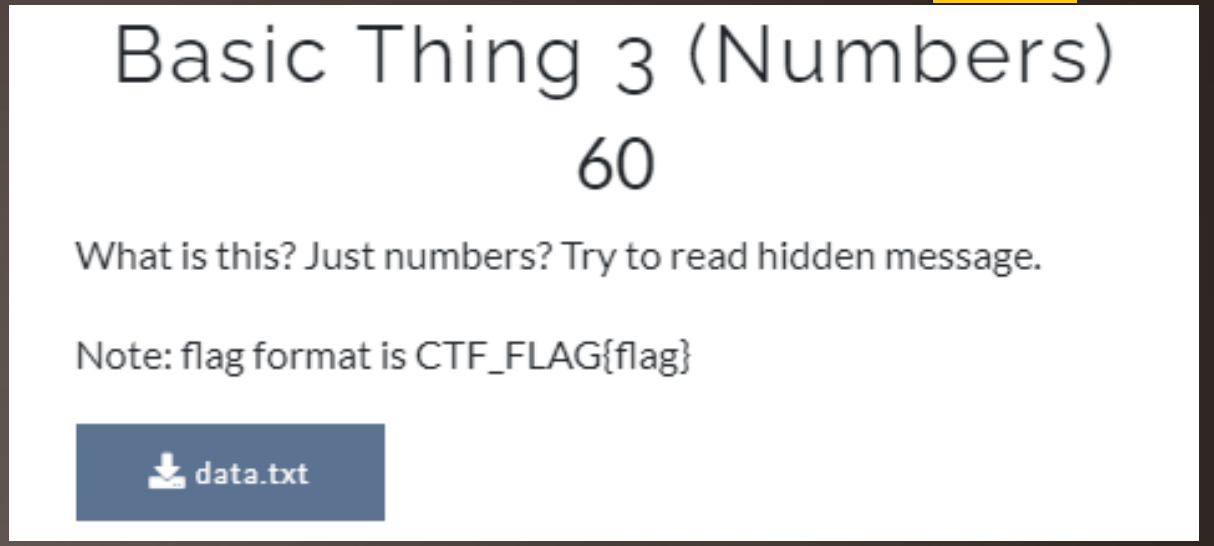
- CTF_FLAG{str4ng3}

Simple Crypto 5

- I converted this numbers to ASCII and used Monoalphabetic Substitution Decoder.

```
n = [80, 71, 83, 95, 83, 89, 78, 84, 123, 82,
     105, 114, 69, 108, 95, 89, 114, 103, 103,
     114, 69, 95, 85, 64, 70, 95, 48, 74, 97,
     95, 65, 104, 122, 79, 51, 69, 125]
# res = [chr(i) for i in n]
# cipher = ''.join(res)
cipher = 'PGS_SYNT{RirEl_YrggrE_U@F_0Ja_AhzO3E}'
```

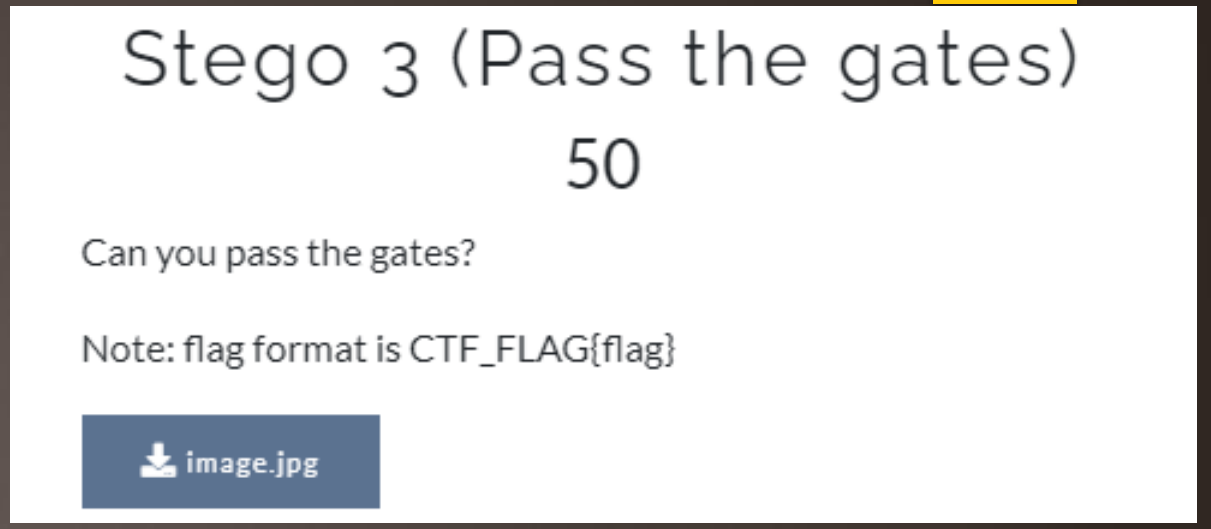
- https://www.dcode.fr/monoalphabetic-substitution



- CTF_FLAG{EveRy_LetteR_H@S_0Wn_NumB3R}

Stego 1

- Firstly, I used this site and found a password “S@n_Fr@nc1sc0” in file’s strings. I understood much later, it wasn’t the flag but exactly password to smth.
- https://stegonline.georgeom.net/image
- Then I checked whether this file was an archive with password and changed file extension to “.jpg.rar”. Nope.
- This site decodes stego files that may have a password.
- https://futureboy.us/stegano/decinput.html



- CTF_FLAG{G0ID3N_G@t3_C@I1f0rn1@}

Stego 2

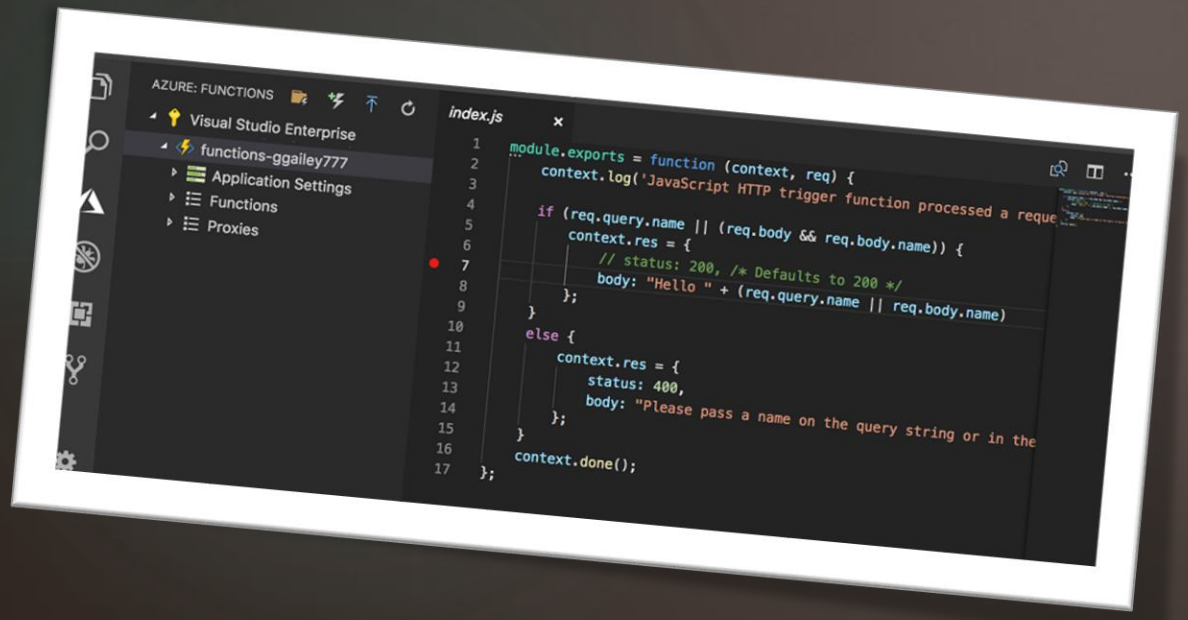
- Once again I used StegOnline and extracted data from the photo, choosing R0, as it was said in the hint.

- https://stegonline.georgeom.net/image

- CTF_FLAG{r0_crew_is_only_significant_thing}

Misc 1

- As it didn't want to be uploaded to StegOnline, I changed file extension to ".jpg" that made upload successful and I found the flag in strings. Profit.
- https://stegonline.georgeom.net/image



- CTF_FLAG{Im@G3s_C@N_H1d3_S3cr3Ts}

Crypto 1

- Using XOR property (if A⊕B=C, then A⊕C=B and C⊕B=A) and some intuition we can obtain:

```
A = 'L[IPICNH'
B = 'CTF_FLAG'
r = [chr(ord(a) ^ ord(b)) for a,b in zip(A, B)]
print(r)
# =>   ['\x0f', '\x0f', '\x0f', '\x0f',
# '\x0f', '\x0f', '\x0f', '\x0f']
# => E = \x0f
H = 'L[IPICNHtL?aH}O{zCO{>?A.r'
E = '\x0f'
res = [chr(ord(h) ^ ord(E)) for h in H]
print(''.join(res))
# => CTF_FLAG{C0nGr@tuL@t10N!}
```

Basic Crypto 4 (Hen ^ EGG)

35

HEN = L[IPICNHtL?aH}O{zCO{>?A.r

EGG = 0f

Note: Flag format is CTF_FLAG{flag}

- CTF_FLAG{C0nGr@tuL@t10N!}

Crypto 2

- The Hint was “Et tu, Brute?”. Of cause, Caesar cipher.
- Shift = 17.
- https://www.boxentriq.com/code-breaking/caesar-cipher

- CTF_FLAG{ItIsNotClassicalCaesarCipher}

Crypto 3

- I found such number d: $d \equiv e^{-1} \pmod{\lambda(n)}$, where $\lambda(n) = lcm(p-1, q-1)$. Then calculated $c^d \equiv (m^e)^d \equiv m \pmod{n}$, where n = p×q.
- https://www.dcode.fr/lcm
- https://ilovecalc.com/calcs/maths/modular-inverse/1306/
- http://upbyte.net/news/dlinnaja_arifmetika_onlajn/2017-03-04-961
- https://planetcalc.ru/8979/
- https://www.rapidtables.com/convert/number/hex-to-ascii.html

- CTF_FLAG{childish_rsa_ins1d3}

Crypto 4

- Once again Base64 and Substitution cipher.
- https://www.base64decode.org
- https://www.dcode.fr/monoalphabetic-substitution

```
U08gWUpCTUdOTEpFTVRCICBFIElIWElHU0dIR1NOTyBZU0
1UVUogU0kgRSBQVUdUT1cgT1YgVU9ZSkJNR1NPTCBYQiBE
VFNZVCBIT1NHSSBOViBNWkVTT0dVQ0cgRUpVIEpVTVpFWV
VXIERTR1QgWVNNVFVKR1VDRyAgRVlZTkpXU09MIEdOIEUg
V1NDVVcgSUJJR1VQICBHVFUgIEhPU0dJICBQRUIgWFUgSV
NPTFpVIFpVR0dVSkkgIEdUVSBQTklHIFlOUFBOTyAgIE1F
U0pJIE5WIFpVR0dVSkkgIEdKU01aVUdJIE5WIFpVR0dVSk
kgIFBTQ0dISlVJIE5WIEdUVSBFWE5GVSAgRU9XIElOIFZO
SkdUICBHVFUgSlVZVVNGVUogV1VZU01UVUpJIEdUVSBHVU
NHIFhCIE1VSlZOSlBTT0wgR1RVIFNPR1VKSVUgSUhYSUdT
R0hHU05PIFlHVl9WWkVMe0dUU0lfTUhBQVpVX11FT09OR1
9YVV9JT1pGVVd9IA==
```

Basic Crypto 3 (Strange attachment)

70

You received email from unknown sender. Someone sent you the letter with encrypted message. Try to read it.

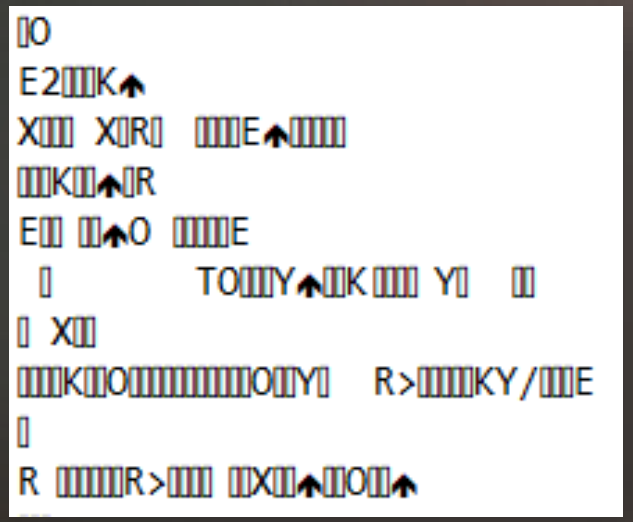
Note: Flag format is CTF_FLAG{text}

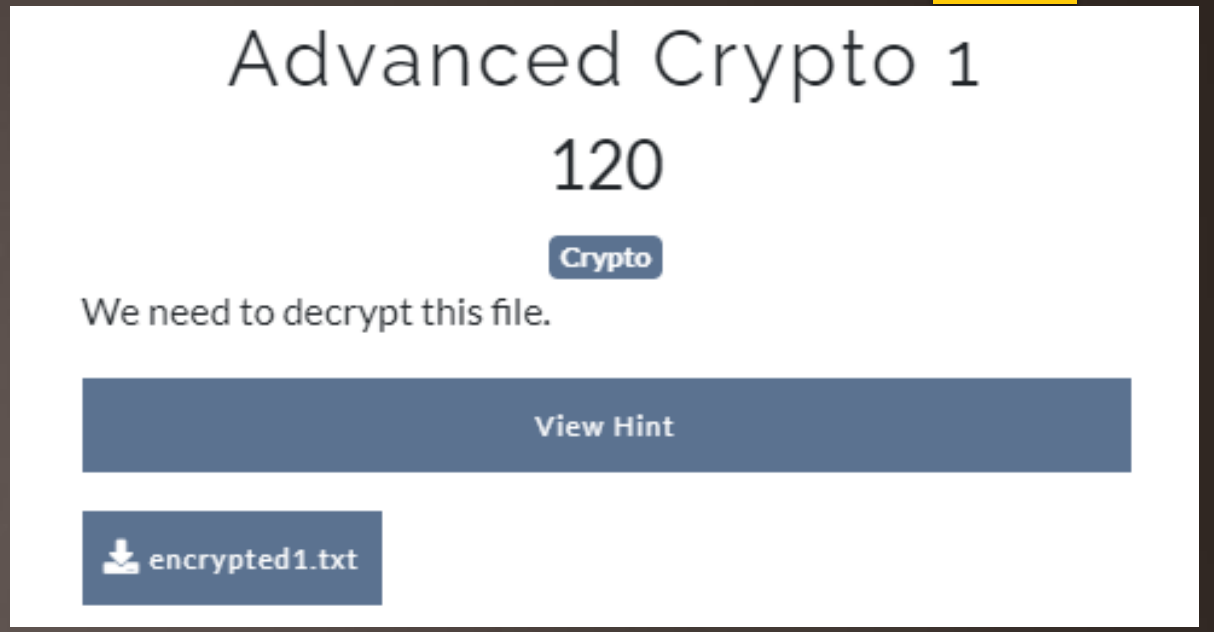
1-text.txt

- CTF_FLAG{THIS_PUZZLE_CANNOT_BE_SOLVED}

Crypto 5

- There were a lot of symbols, so I decided to do frequency analysis, but it was unsuccessful. Then I have tried to change encodings, use different ciphers…
- The solution was XOR.
- https://wiremask.eu/tools/xor-cracker/





- CTF_FLAG{xorIsNotSoSecureAsItMightSeems}

OSINT 1

- Firstly, I was looking for country, where Euronics can be. It's everywhere! OK.
- Then I saw reflection of SONAR and found this beauty salon in Google Maps.

OSINT 1 (Coffee time)

50

You want to become the FBI special agent, but firstly we need to check if you're ready for the real tasks ;) So your first task is to find the meeting point of two suspects. FBI department has only the photo and the mobile message with next text "Let's have a coffee in a cafe opposite this place".

Note: use underscore symbol between words, all letters are in the lowercase.

OSINT-1.png

- CTF_FLAG{costa_coffee}

OSINT 2

- Tried to find him on Facebook and LinkedIn. Finally, I found him in Twitter. There was a link to his GitHub.
- https://twitter.com/clarkmartin1981/with_replies
- https://github.com/mj1981clark
- There are only few recent commits, so I'd checked them and found a link to PasteBin. To unlock the paste I had to find password, what was hard enough.
- There was deleted GIF "barcode.gif" in GitHub. It was a dotcode with encrypted password.
- https://products.aspose.app/barcode/recognize/dotcode

OSINT 2 (Secret)

100

FBI department got hard task to find the secret on the one of text storage sites stored by hacker. The only information we have about this person:

- his name is Martin Clark
- he was born in 1981
- he is programmer
- his nicknames: clark1981martin, clarkmartin1981, martinclark1981, martin1981clark
- he likes social networks

- CTF_FLAG{C0ngrats!YouF1ndMySecr3t}

OSINT 3

- https://www.zoomeye.org
- Searched:

country:"au" +after:"2015-01-01" +before:"2016-01-01" +port:"80" +app:"Drupal cms"

OSINT 3 (Server)

70

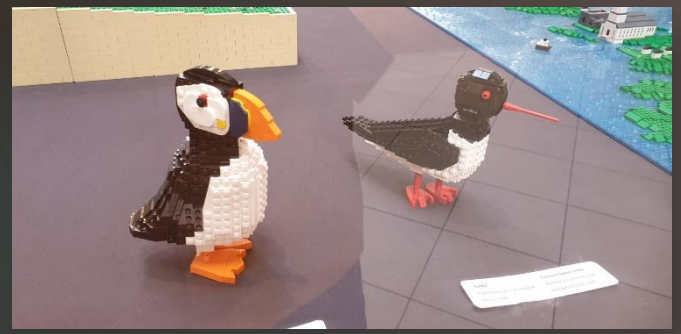
Find the IP address of the Australian Drupal server with opened port 80 that was Internet scanned in 2015.

Note: Flag format is CTF_FLAG{IP}

- CTF_FLAG{203.89.197.236}

OSINT 4

- There are puffin and oystercatcher on this photo. I found lego bird exhibition in Sandwich, USA and puffin festival in North Berwick, Scotland, but these answers were wrong.



- Then I found photo of this oystercatcher in Pinterest and page of its creator, who wrote this article:
- https://dagur.fo/si-kirkjubomurin-glasir-og-cristianskirkjuna-sum-lego-bygningar

OSINT 4 (Exhibition)

200

During pizza party, your friend talked about his hobby: in his free time he collects unusual figures from Lego parts. He recently saw a photo of his favorite birds with a mention of a planned exhibition, but no address was given. Knowing about your searching abilities, friend asked for help in finding the city and name of the center where the exhibition will be held.

Note: flag format is CTF_FLAG{City_Center_Name}, use underscore symbol between words, words are in English.

OSINT-4.jpg

- CTF_FLAG{Torshavn_SMS_Center}

9-th task from rook

Dwarfs really loved algebra. For centuries they tried to invent a robust homomorphic encryption.
They left you the following hints.

1. F1 = GF(2)[x]/(x^8+x^7+x^5+x^4+1) - be the space of cleartexts
2. F2 = GF(2)[x]/(x^8+x^7+x^2+x+1) - be the space of ciphertexts

3. Enc: F1 -> F2 - be encryption, which is some map that is
   - homomorphism of finite fields
   - bijection (so you can correctly encrypt and decrypt any text)
   - maps 0x5c = x^6 + x^4 + x^3 + x^2 into 0xf9 = x^7 + x^6 + x^5 + x^4 + x^3 + 1

Decrypt the secret message from the past:

84 B0 DE 09 58 C7 21 53 C7 CE 09 21 3D C7 09 EE C7 0E 09 CE CD 9A B0 FF 9A B0 FF DA

```python
import numpy as np

enc_hex = ['0x84', '0xB0', '0xDE', '0x09', '0x58', '0xC7', '0x21', '0x53', '0xC7', '0xCE', '0x09', '0x21', '0x3D', '0xC7',
    '0x09', '0xEE', '0xC7', '0x0E', '0x09', '0xCE', '0xCD', '0x9A', '0xB0', '0xFF', '0x9A', '0xB0', '0xFF', '0xDA']

enc_bin = ['10000100', '10110000', '11011110', '00001001', '01011000', '11000111', '00100001', '01010011', '11000111',
    '11001110', '00001001', '00100001', '00111101', '11000111', '00001001', '11101110', '11000111', '00001110', '00001001',
    '11001110', '11001101', '10011010', '10110000', '11111111', '10011010', '10110000', '11111111', '11011010']

# for i in a:
#     k = list(i)
#     x.append('x^7'*int(k[0])+'+x^6'*int(k[1])+'+x^5'*int(k[2])+'+x^4'*int(k[3])+'+x^3'*int(k[4])+'+x^2'*int(k[5])+
#     '+x'*int(k[6])+'+1'*int(k[7]))

# for faster input to SageMathCell
enc_x = ['x^7+x^2', 'x^7+x^5+x^4', 'x^7+x^6+x^4+x^3+x^2+x', 'x^3+1', 'x^6+x^4+x^3', 'x^7+x^6+x^2+x+1', 'x^5+1',
    'x^6+x^4+x+1', 'x^7+x^6+x^2+x+1', 'x^7+x^6+x^3+x^2+x', 'x^3+1', 'x^5+1', 'x^5+x^4+x^3+x^2+1',
    'x^7+x^6+x^2+x+1', 'x^3+1', 'x^7+x^6+x^5+x^3+x^2+x', 'x^7+x^6+x^2+x+1', 'x^3+x^2+x', 'x^3+1',
    'x^7+x^6+x^3+x^2+x', 'x^7+x^6+x^3+x^2+1', 'x^7+x^4+x^3+x', 'x^7+x^5+x^4', 'x^7+x^6+x^5+x^4+x^3+x^2+x+1',
    'x^7+x^4+x^3+x', 'x^7+x^5+x^4', 'x^7+x^6+x^5+x^4+x^3+x^2+x+1', 'x^7+x^6+x^4+x^3+x']

enc_pow = [89, 192, 17, 205, 191, 204, 87, 138, 204, 48, 205, 87, 13, 204, 205, 248, 204, 107, 205, 48, 31, 118,
    192, 183, 118, 192, 183, 239] # found with SageMathCell
```

```python
# for j in range(1, 256):
#     if np.gcd(j, 255) == 1 and (135 * j) % 255 == 225:
#         gen_els.append(j)
# print(gen_els)

# all mul gen (powers of a2) of F2 that satisfy 0x5c -> 0xf9
gen_els = [13, 47, 64, 98, 149, 166, 217, 251]

# function a^n in F1
def powerx(a, n):
    fac1 = np.array([1, 1, 0, 1, 1, 0, 0, 0, 1])
    m = np.array(a)
    for i in range(n - 1):
        m = np.polydiv(np.polymul(m, a), fac1)[1] % 2
    return m

# mul gen in F1
a1 = np.array([1, 0, 1, 1])

for i in gen_els:
    el = powerx(a1, i)
    c = []
    for j in enc_pow:
        c.append(hex(int(''.join([str(int(y)) for y in powerx(el, j)]), 2)))
    msg = []
    for k in c:
        msg.append(chr(int(k, 16)))
    print(''.join(msg))
```

SageMathCell

Type some Sage code below and press Evaluate.

```
_.<X> = GF(2)[]
K.<x> = GF(2^8, modulus=X^8+X^7+X^5+X^4+1)
K.multiplicative_generator()
```

```
x^3 + x + 1
```

Type some Sage code below and press Evaluate.

```
 _.<X> = GF(2)[]
K.<x> = GF(2^8, modulus=X^8+X^7+X^5+X^4+1)
K.multiplicative_generator()
((x^3 + x + 1)^13).multiplicative_order()
a = (x^3 + x + 1)^13
b = x^6 + x^4 + x^3 + x^2
log(b, a)
```

Evaluate

```
135
```

- And that's the result

1. æ> BZ&Zü &¦Z KZÑ üM#M#ö
2. í¿Cå&øC³CêCøªà¿âà¿âo
3. And Ieaves are yet swinging,
4. å[CíZ&ZüC&ZCZßCü°¼#¼#
5. nóC§eavesCaeC-eCs-ng-ng
6. I¥¨ A©va©! vD© ¡©Å !%5¥
7. B¿? æ&ø Q õ« øÙââÖ
8. §¥òC©va©!CvÔ©C©`C!Ý$¥

Thank you!